Integrated Simulation and Model-Checking for the Analysis of Biochemical Systems

Ciocchetta, F., Gilmore, S., Guerriero, M. L., & Hillston, J. (2009). Integrated Simulation and Model-Checking for the Analysis of Biochemical Systems. Electronic Notes in Theoretical Computer Science, 232, 17-38. 10.1016/j.entcs.2009.02.048

Integrated Simulation and Model-Checking for the Analysis of Biochemical Systems

Federica Ciocchetta[a,1], Stephen Gilmore[a,2], Maria Luisa Guerriero[a,3], Jane Hillston[a,b,4]

[a] *Laboratory for Foundations of Computer Science, The University of Edinburgh, EH8 9AB Edinburgh, Scotland*

[b] *Centre for Systems Biology at Edinburgh (CSBE), Edinburgh, Scotland*[5]

Abstract

Model-checking can provide valuable insight into the behaviour of biochemical systems, answering quantitative queries which are more difficult to answer using stochastic simulation alone. However, model-checking is a computationally intensive technique which can become infeasible if the system under consideration is too large. Moreover, the finite nature of the state representation used means that *a priori* bounds must be set for the numbers of molecules of each species to be observed in the system.

In this paper we present an approach which addresses these problems by using stochastic simulation and the PRISM model checker in tandem. The stochastic simulation identifies reasonable bounds for molecular populations in the context of the considered experiment. These bounds are used to parameterise the PRISM model and limit its state space. A simulation pre-run identifies interesting time intervals on which model-checking should focus, if this information is not available from experimental data.

Keywords: Systems biology, process algebra, model-checking, stochastic simulation

1 Introduction

Model-checking and stochastic simulation techniques have both been applied to the study of biochemical systems, and both allow researchers to make predictions and test hypotheses. The questions which they answer can be different and complementary.

The stochastic simulation approach allows modellers to analyse the time evolution of all species composing a system at the same time. However, since one simulation run generates a single trajectory out of all the possible behaviours of systems, usually average values among several runs need to be considered to achieve the necessary level of confidence in the results obtained.

[1] Email: fciocche@inf.ed.ac.uk

[2] Email: stg@inf.ed.ac.uk

[3] Email: mguerrie@inf.ed.ac.uk

[4] Email: jeh@inf.ed.ac.uk

[5] The Centre for Systems Biology at Edinburgh is a Centre for Integrative Systems Biology (CISB) funded by the BBSRC and EPSRC in 2006

Probabilistic model-checking instead answers quantitative temporal queries by performing an exhaustive exploration of all the possible paths through the system. Model-checking requires both the model and the specification of the system under study to be formally specified: this allows the user to detect possible errors in the model or the presence of deadlock states, and to automatically verify whether or not relevant properties are satisfied by the model. For these reasons, when analysing biochemical systems, it is often more desirable to perform model-checking than simulation. Unfortunately, model-checking has one major problem, the *state-space explosion*: a system with too many states becomes intractable and needs to be constrained.

The aim of the present work is to investigate how combined use of stochastic simulation and model-checking can lead to a better understanding of biochemical systems. In particular, we investigate how to exploit the knowledge acquired from the simulation to make the model-checking feasible. Specifically, the simulation results are used to establish reasonable lower and upper bounds for the molecule counts of the species involved. We show with a simple example how, by using the bounds estimated by our approach, we are able to substantially speed up the model-checking without introducing significant error.

We use the high-level modelling language Bio-PEPA [6,7], a timed process algebra designed specifically for the description of biological phenomena and their analysis through quantitative methods such as stochastic simulation and model-checking. Several alternative representations can be automatically generated from a Bio-PEPA specification, allowing us to perform different kinds of analyses on the same model. In particular, we consider here two generated models: one suitable for stochastic simulation using Dizzy [17], the other suitable for model-checking using PRISM [16].

The novel contribution offered by the present paper is the use of simulation and model-checking conjoined in two ways. Firstly, we use simulation to bound the model-checking problem and later we compare model-checking results obtained through both exact and approximate probabilistic model-checking. The former method elaborates the full state-space of the model and uses linear algebra to solve the underlying Markov chain. The latter uses simulation to answer the model-checking problem, up to a satisfactory confidence interval.

The rest of the paper is structured as follows. In Sect. 2 we discuss some motivations of our work and we illustrate them by means of a simple example. Sect. 3 is devoted to the description of the background. In Sect. 4 we present our approach and in the following Sect. 5 we apply it to two biochemical networks. Some related work is reported in Sect. 6 and finally, some concluding remarks are presented in Sect. 7.

2 Motivation

Analysing models of biological processes via probabilistic model-checking has considerable appeal. As with stochastic simulation the answers which are returned from model-checking give a thorough stochastic treatment of the small-scale phenomena which are of greatest interest to computational biologists today. However, in contrast to a simulation run which generates just one of many possible trajectories, the analysis results computed by probabilistic model-checking give a definitive answer. That is, it is not necessary to re-run the analysis repeatedly and compute ensemble averages of the results. Further, by building a reward structure over the model it is possible to express complex analysis questions

and evaluate these through model-checking. This form of analysis has the power to expose of the system under study significant temporal behaviour which could not be appreciated from simple inspection of the species time-series generated by simulation runs (see for example [9]).

Set against this, the probabilistic model-checking approach faces the well-known problem of state-space explosion where, as the complexity of the system under study increases, there is an exponential growth in the state-space of the underlying model. The use of an exact discrete-state representation of the state-space of the model restricts the use of probabilistic model-checking to the analysis of problems where all of the species are available in low copy numbers. Multi-scale models (where some species are in plentiful supply and others have very low molecule counts) generally give rise to discrete-state problems whose numerical solution is infeasible.

Even in the case where all of the chemical species involved are present only in low copy numbers it is still necessary to place a bound on the maximum molecule count which each species will attain. For models involving biochemical processes such as synthesis, no such bounds can be established. In the present paper we describe the application of stochastic simulation to the problem of bounding discrete-state models allowing us to convert an unbounded model-checking problem into a bounded one.

To illustrate the problem which we are discussing here consider a simple model of the Michaelis-Menten reactions involving four chemical species: an enzyme E, a substrate S, a compound $E{:}S$ and a product P. The species react over three reaction channels: r_1 converting E and S to $E{:}S$, the backward reaction r_{-1} taking $E{:}S$ to E and S, and the reaction r_2 converting the compound $E{:}S$ into product P and releasing the enzyme E. The reaction rates are governed by kinetic laws involving rate constants (k_1, k_{-1} and k_2) and the molecular counts of the species involved.

The chemical equation describing Michaelis-Menten reactions is

$$E + S \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} E{:}S \xrightarrow{k_2} E + P \ .$$

In the Bio-PEPA language the notation f_{r_i} is used to indicate the rate associated with the reaction r_i.

$$f_{r_1} = k_1 \times E \times S$$
$$f_{r_{-1}} = k_{-1} \times E{:}S$$
$$f_{r_2} = k_2 \times E{:}S$$

When initiated with low molecule counts $(E, S, E{:}S, P) = (5, 5, 0, 0)$ this model gives rise to a state space of very modest size, as indicated in Fig. 1. Starting in the state $(5, 5, 0, 0)$ each of the four species E, S, $E{:}S$ and P can achieve molecular counts in the bounded integer range 0 to 5. Of the $6 \times 6 \times 6 \times 6 = 1296$ potential states in the full product state space of this model only 21 of these are actually reachable by any sequence of reactions. One reachable state, $(5, 0, 0, 5)$, is a "deadlock" state with no outgoing transitions. Reaction r_1 is prevented because $S = 0$ and reactions r_{-1} and r_2 are prevented because $E{:}S = 0$.

However, if we consider an extension of the model with an additional reaction r_0 which

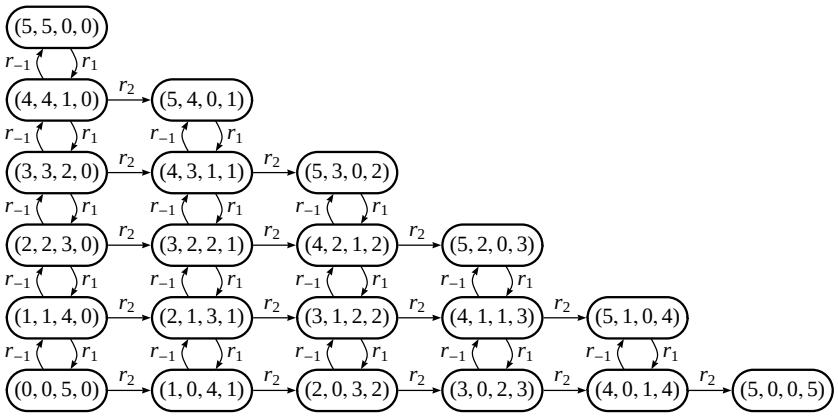


Fig. 1. Discrete state-space representation of the Michaelis-Menten reactions.

synthesises the compound *E:S*

$$\emptyset \xrightarrow{k_0} E{:}S$$

with the synthesis occurring at a constant rate $f_{r_0} = k_0$ then this additional reaction channel changes the analysis of the model dramatically. The state which was previously a deadlock state now admits an r_0 reaction which leads it to a previously unreachable state, $(5, 0, 1, 5)$. The reactions r_{-1}, r_1 and r_2 can occur in states reachable from that one as shown in Fig. 2.

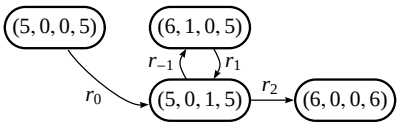


Fig. 2. Synthesis of the compound adds further states.

Each of these states, *and every other state*, now allows an r_0 reaction, taking them to previously unreachable states each of which allows r_0 and reactions r_{-1}, r_1 and r_2 subsequent to that. The effect of introducing this single synthesis reaction is that we now cannot find any upper bound N such that the molecular species counts are guaranteed to lie in the bounded integer range 0 to N. In general, if we are unable to bound the reachable state-space then we cannot analyse our model by probabilistic model-checking.

Here we seek to bring the unbounded state-space back within bounds by exploiting the following observations.

(i) The generation of the derivation graph of the underlying state-space does not take into account the numerical values assigned to the rate constants, and the propensity functions which depend on those. This means that the derivation graph may include many states which the system is almost sure not to reach within a particular time bound.

(ii) Most chemical systems involve several widely varying time scales, so such systems are nearly always stiff [18]. A consequence of this is that the first passage time to many states is likely to be long and truncation of the state-space using a time-bounded

reachability metric is likely to be productive.

(iii) Many of the logical formulae which we wish to check involve reaching within a fixed time bound model states which satisfy a given predicate.

(iv) Stochastic simulation methods such as Gillespie's Direct Method [8] generate exact stochastic simulations of trajectories from the initial state to states reachable within a given time bound.

3 Background

The context of application we consider is that of biochemical networks. A biochemical network is composed of n species which interact through m reactions; the dynamics of reaction j is described by a kinetic law f_j. The quantitative behaviour of a biochemical network depends on the initial values of the involved species and on the kinetic parameters.

In the following, we distinguish between *structurally bounded* and *structurally unbounded* biochemical networks. A biochemical network is *(structurally) bounded* if for each species S_i ($i \in \{1, \ldots, n\}$) there exists a value $Max_i \in \mathbb{N}$ such that $X_i(t) < Max_i \; \forall t$, where $X_i(t)$ is the amount of species i at time t. The values Max_i for a generic biochemical network, if they exist, depend on the kind of reactions involved, on their kinetic laws and constants, and on the initial state of the system. If, instead, the amount of one or more species has the potential to grow without bound, the network is *structurally unbounded*.

If we assume a finite number of molecules for each species in the initial state, and we do not consider reactions which can increase the amount of any species, then we have structurally bounded networks. If instead, we consider synthesis reactions (e.g. $\emptyset \rightarrow A$, $C \rightarrow C + A$) or arbitrary split reactions, we have structurally unbounded networks. In many cases a structurally unbounded network may have a pragmatic bound because of the quantitative relations between the molecules and the reactions composing the network. Specifically, even though some synthesis reactions are present, the average value of each species can be bounded.

Note that, in the real world, biochemical networks are generally bounded: degradation, for instance, is an important mechanism cells use to avoid an uncontrolled increase of molecules. However, structurally unbounded networks are interesting because an anomalous behaviour of some biological entity could trigger such an uncontrolled growth. Moreover, biological models are often limited to particular sub-systems, in which the bounding reaction might not be included.

3.1 Bio-PEPA

The Bio-PEPA language [7,6] allows us to explicitly represent some features of biochemical models, such as the stoichiometry of reactions and the role of each species in a given reaction, and allows the definition of general kinetic laws. Bio-PEPA models can be analysed by different techniques (stochastic simulation, analysis based on ODEs, numerical solution of the continuous-time Markov chain (CTMC), and probabilistic model-checking), since the mappings of Bio-PEPA models into specifications for those approaches have been defined [6].

The language is based on discrete levels of parameterised species: each component represents a species and its parameter may be interpreted as the number of molecules or

discrete levels of concentration depending on the type of analysis to be applied. Parametric levels are considered for the definition of the transition system and for the derivation of a CTMC whose states represent the concentration levels of the species.

The syntax of Bio-PEPA is defined as:

$$S ::= (\alpha, \kappa)\ \mathsf{op}\ S \mid S + S \mid C \qquad P ::= P \underset{\mathcal{L}}{\bowtie} P \mid S(l)$$

where $\mathsf{op} = \downarrow \mid \uparrow \mid \oplus \mid \ominus \mid \odot$.

The component S is called a *sequential component* (or *species component*) and represents the species whereas the component P, called a *model component*, describes the system and the interactions among components. The parameter $l \in \mathbb{N}$ represents the discrete level of concentration. The prefix term $(\alpha, \kappa)\ \mathsf{op}\ S$ contains information about the role of the species in the reaction associated with the action type α: κ is the *stoichiometry coefficient* of the species and the *prefix combinator* "op" represents the role of the element in the reaction. Specifically, $\downarrow$ indicates a *reactant*, $\uparrow$ a *product*, $\oplus$ an *activator*, $\ominus$ an *inhibitor* and $\odot$ a generic *modifier*. The operator "+" expresses the choice between possible actions and the constant C is defined by an equation $C \overset{def}{=} S$. Finally, the process $P \underset{\mathcal{L}}{\bowtie} Q$ denotes the cooperation between components: the set $\mathcal{L}$ determines those activities on which the operands are forced to synchronise.

In order to specify a model in Bio-PEPA, in addition to the definition of the species and model components, we need to define a set of functional rates $\mathcal{F}$ expressing the kinetic laws of the reactions, a set of constant parameters $\mathcal{K}$ and the compartment size (in the set $\mathcal{V}$). For discrete state space analysis the behaviour of the system is defined in terms of an operational semantics. The rules are reported in [6]. In the following we indicate a Bio-PEPA model with $\mathcal{M}$.

The Bio-PEPA language is supported by software tools which automatically process Bio-PEPA models and generate other representations in forms suitable for simulation and model-checking. The generated simulation model can be executed using the Dizzy stochastic simulator [17]. The representation which is used for discrete state-space generation and analysis by numerical solution of the underlying CTMC is expressed in the reactive modules language supported by the PRISM model-checker. In addition the Bio-PEPA tools generate reward structures and common CSL formulae used in model-checking.

In this paper we consider only numbers of molecules, therefore the CTMCs obtained from our Bio-PEPA models are in terms of number of molecules as well.

3.2 Model Analysis

Both stochastic simulation and the probabilistic model-checking that we consider are based on an underlying mathematical model which is a CTMC. A continuous-time Markov chain is a discrete-state process whose evolution is governed by exponential distributions, giving the stochastic process the *memoryless* or *Markovian* property.

Gillespie's stochastic simulation algorithm [8] is a widely-used method for the simulation of biochemical systems. It applies to homogeneous, well-stirred systems in thermal equilibrium and constant volume. Broadly speaking, the goal is to describe the evolution of the system $\mathbf{X}(t)$, described in terms of the number of elements of each species, starting from an initial state.

PRISM [16] is a probabilistic model checker, which can be used to check properties of discrete-time Markov chains and Markov decision processes, in addition to CTMCs. It has been used to analyse systems from a wide range of application domains. Models are described using the state-based PRISM language and for a CTMC model it is possible to specify quantitative properties of the system using a temporal logic, called *CSL* [1,2] (Continuous Stochastic Logic).

The PRISM language is composed of *modules* and *variables*. A model is composed of a number of modules which can interact with each other. From a Bio-PEPA model there will be one module for each species. A module contains a number of local variables. The values of these variables at any given time constitute the state of the module. Such a variable will be used to record the number of molecules which are currently present in the system. The global state of the whole model is determined by the local state of all modules. This corresponds to $\mathbf{X}(t)$ in the stochastic simulation. The behaviour of each module is described by a set of guarded commands. Each command describes a transition which the module can make if the guard is true. A command includes an update which gives new values to the variables. In the mapping from a Bio-PEPA model the transitions correspond to the activities of the Bio-PEPA model and the updates take the stoichiometry into account. Transition rates are specified in an auxiliary module which defines the functional rates corresponding to all the reactions.

The well-formed formulae of CSL are made up of *state formulae* ϕ and *path formulae* ψ. The syntax of CSL is below.

$$\phi ::= \mathit{true} \mid \mathit{false} \mid a \mid \phi \wedge \phi \mid \phi \vee \phi \mid \neg\phi \mid \mathcal{P}_{\bowtie p}[\psi] \mid \mathcal{S}_{\bowtie p}[\phi]$$

$$\psi ::= \mathrm{X}\,\phi \mid \phi\ \mathrm{U}^{I}\ \phi \mid \phi\ \mathrm{U}\ \phi$$

Here a is an atomic proposition, $\bowtie \in \{<, \leq, >, \geq\}$ is a relational parameter, $p \in [0,1]$ is a probability, and I is an interval of $\mathbb{R}^+$. The operator $\mathcal{P}_{\bowtie p}[\psi]$ is used to express transient properties (i.e. dependent on time) whereas the operator $\mathcal{S}_{\bowtie p}[\phi]$ is used to express steady state properties (i.e. hold in the long run). The operators X and U are used to express neXt and Until properties, respectively. Time-bounded Until formulae U^I are indexed by an interval I. Derived logical operators such as implication ($\Rightarrow$) can be encoded in the usual way.

3.3 Model-checking with PRISM

PRISM [16] includes support for the specification and analysis of properties based on rewards: real values are associated with certain states or transitions of the model. In this way it is possible to reason about various quantitative measures such as "expected number of processes/proteins" or "expected number of reactions". The PRISM reward language allows the expression of both instantaneous and cumulative rewards.

PRISM supports both exact and *approximate* probabilistic model-checking (in the style of the APMC tool [12,13]). In approximate model-checking Monte-Carlo simulation is used together with the theory of randomised approximation schemes to give accurate approximations of satisfaction probabilities. Properties of large discrete-state systems can be checked using very little memory but in practice the run-times of such simulations can be very long. In our experience a simulation pre-run followed by exact probabilistic model-

checking is less costly than computing the same results using approximate model-checking alone. We will compare the results obtained from the two methods.

4 Estimating Lower and Upper Bounds on Molecules

As mentioned above, from a Bio-PEPA system $\mathcal{M}$ we can generate a Dizzy model for stochastic simulation and a PRISM model for model-checking. The initial amount of each species, stoichiometric information and the kinetic laws with the associated parameters are needed for the simulation model. This information can be collected from $\mathcal{M}$. In addition, the lower and upper bounds for each species are also needed in the PRISM model in order to build a finite CTMC and, hence, to make the analysis by means of CTMC feasible. Especially in the case of unbounded networks it is essential to define an upper bound that makes the analysis feasible but still is able to capture the behaviours of interest of the system.

The main issue we investigate in this work is: how to specify the minimum and maximum amount of each involved species? In principle, in some networks, bounds on the number of molecules can be obtained from pre-existing biological knowledge about the system and from experimental data. However, this information is often incomplete, we could have only little pre-existing knowledge of the (normal and anomalous) behaviour of the system and there can be a high variability among different experiments. In these cases the derivation of the bounds is particularly hard to face. Furthermore, the bound values are tightly dependent on the initial conditions and on the parameter values; since wet experiments are generally time-consuming and costly, assuming that such bounds are known for each relevant parameter set is not realistic.

Even in the complete absence of experimental data it is possible for a structurally bounded network to derive theoretically both lower and upper bounds for the number of molecules. For instance, in the simple case with stoichiometry equal to one and no arbitrary split of molecules/complexes, the lower bound can be fixed to 0, while an upper bound is given by the sum of the initial amount of each species.

However, in general, for real life complex systems, it is hard to derive this information and, depending on the relative rates of the reactions, the theoretical bounds could be practically unreachable. Unfortunately, bounds calculated in this way are very loose and the system is almost always intractable for model-checking if these bounds are used. Furthermore, when unbounded networks are considered it is not even possible to derive these loose theoretical bounds.

Here we use stochastic simulation to estimate the minimum and maximum number of molecules for each species. We run a number of simulation experiments and use the output results as a rule of thumb for selecting lower and upper bounds for model-checking. The number of simulation runs should be chosen depending on the variability of the specific system under study. Due to the nature of stochastic simulation, the more simulation experiments we perform, the higher will be our confidence in the derived bounds. This approach is the sole way of deriving bounds if the total number of molecules present in the system can increase compared to the initial state.

In the case of structurally bounded networks, the species can assume values between a minimum and a maximum value. From the simulation results, we can derive an estimate of the maximum value for each species S_i as $Max_i = \max\{X_i^j(t),\ j = 1, \ldots, N_{runs},\ t \in [0, T]\}$,

where $X_i^j(t)$ is the amount of the species i, in the simulation j at time t, N_{runs} is the number of simulation runs and T is the simulation stop-time, which depends on the specific network and is usually defined according to experimental data or set to some time of interest. A similar approach is used to derive the minimum value Min_i.

In structurally unbounded networks it is possible that a species does not have an upper bound on the number of molecules on the long run. However here we are interested in the behaviour of the system until a fixed time T of interest and therefore we consider the maximum value for each species in the interval $[0, T]$. By considering the simulation results, we can derive an estimate of the maximum and of the minimum value as before. It is worth noting that, by this approach, we impose some bounds on systems which are not bounded. As a consequence, we can only verify time-bounded formulae over the interval $[0, T]$. In practice this is not a severe restriction because almost all of the formulae which we use are time-bounded Until formulae.

A final observation is about the time interval $[0, T]$. In our experience *transient* properties (i.e. dependent on time) are of greatest interest to biologists, not steady-state behaviour. Generally, the time bound considered is the one used in the experimental work. When analysing models, therefore, we are interested in checking properties within that time bound, as these can be validated against the available experimental data. However, when experimental data is not available or is partial or incomplete, the time bound needs to be arbitrarily defined. Since checking properties can be very time-consuming for models representing real life systems, it is desirable to focus on the shortest time interval which allows us to capture the interesting behaviour. For example, if a steady state exists, it is pointless to consider a time longer than the one at which the steady state is reached. On the other hand, one might not be interested in the very first time steps. Again, the time-series generated by a simulation pre-run can provide a good estimate of the time interval to choose for the verification of transient properties.

5 Application and Results

We consider here two simple models in order to illustrate our approach, to show the computational advantage of using the estimated bounds in model-checking, and to discuss the error which is introduced by truncating the state space. These models are abstract representations, under different assumptions, of a general genetic network with a negative feedback. An example of this kind of network is the control circuit for the λ repressor protein CI of λ-phage in *E.Coli*, modelled in [3].

A schema of the general network is reported in Fig. 3. We have four biochemical entities that interact with each other through six reactions. The biochemical entities are the DNA (D), the mRNA (M), a protein in monomeric form (P) and a protein in dimeric form ($P2$). The first reaction in the network is the *transcription* of the mRNA from the DNA. The protein in dimeric form, which is the final product of the network, has an inhibitory effect on the transcription process. The second reaction is the *translation* of the protein from the mRNA. Reactions *degradation_M* and *degradation_P* represent the possible degradation of mRNA and of the protein, respectively. Finally, *dimerization* and *monomerization* are the protein dimerization and its inverse reaction. All reactions are described by mass-action kinetics, apart from *transcription*, which follows Michaelis-Menten kinetics.

The network is structurally unbounded, since both transcription and translation lead

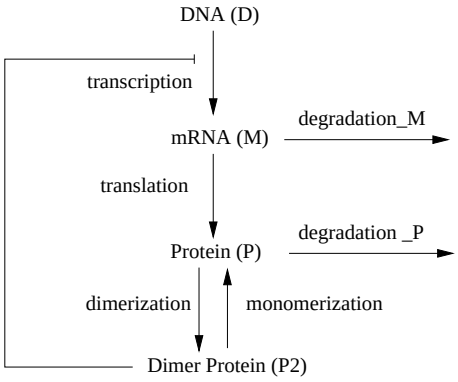


Fig. 3. Genetic network model.

to the creation of new molecules. However, the two degradation reactions and the transcription inhibition by means of the dimeric protein have a regulatory effect on the protein synthesis and therefore, under some conditions, all the species reach a finite average value.

Our two models represent the network described above with two different sets of parameters and a different assumption on the degradation of the protein. The set of parameters used in the first model makes the protein degradation fast enough to yield to a pragmatically bounded system (on average). In the second model, instead, we consider different values for some of the parameters and the complete absence of protein degradation. These assumptions have a dramatic effect on the systems behaviour: it makes the amount of protein increase indefinitely.

5.1 Specification of the Networks in Bio-PEPA

5.1.1 The Network with Protein Degradation ($\mathcal{M}_1$)

In the following we define the Bio-PEPA system $\mathcal{M}_1$ representing the first network.

The set of species components and the model component are defined as follows.

$$D \stackrel{def}{=} (transcription, 1) \odot D;$$

$$M \stackrel{def}{=} (transcription, 1)\uparrow M + (translation, 1) \odot M + (degradation\_M, 1)\downarrow M;$$

$$P \stackrel{def}{=} (translation, 1)\uparrow P + (dimerization, 2)\downarrow P + (monomerization, 2)\uparrow P + (degradation\_P, 1)\downarrow P;$$

$$P2 \stackrel{def}{=} (transcription, 1) \ominus P2 + (dimerization, 1)\uparrow P2 + (monomerization, 1)\downarrow P2;$$

$$Res \stackrel{def}{=} (degradation\_M, 1) \odot Res + (degradation\_P, 1) \odot Res;$$

$$(((D(1) \underset{\{transcription\}}{\bowtie} M(0)) \underset{\{translation\}}{\bowtie} P(0)) \underset{\{dimerization,monomerization\}}{\bowtie} P2(0)) \underset{\{degradation\_M,degradation\_P\}}{\bowtie} Res(1)$$

All the species are in the same compartment, defined as $v_{cell} : 1\ \mathrm{nM}^{-1}$. Initially we have one molecule of DNA and one molecule of the generic modifier *Res*. We omit the information about levels, since in this work we consider the molecular level, as usual in stochastic simulation. The set of functional rates $\mathcal{F}_R$ and stochastic parameters $\mathcal{K}$ are reported below.

$$f_{transcription} = \frac{v \times D}{K_{\mathrm{M}} + P2}; \qquad f_{translation} = k_2 \times M;$$

$$f_{degradation\_M} = k_3 \times M; \qquad f_{degradation\_P} = k_4 \times P;$$

$$f_{dimerization} = \frac{k_5 \times P \times (P-1)}{2}; \qquad f_{monomerization} = k_{-5} \times P2;$$

$$K_{\mathrm{M}} = 356 \text{ molecules}; \quad v = 2.19 \text{ s}^{-1}; \quad k_2 = 0.043 \text{ s}^{-1};$$
$$k_3 = 0.039 \text{ s}^{-1}; \quad k_4 = 0.0007 \text{ s}^{-1};$$
$$k_5 = 0.025 \text{ s}^{-1}; \quad k_{-5} = 0.5 \text{ s}^{-1} \ .$$

5.1.2 The Network Without Protein Degradation ($\mathcal{M}_2$)

The definition of the Bio-PEPA system $\mathcal{M}_2$ representing the second network is very similar to the case of the first network. Below we report only the parts of the specification of $\mathcal{M}_2$ that differ from $\mathcal{M}_1$. The changes concern the definition of the species components P and *Res* (the term for protein degradation is removed), the elimination of the functional rate $f_{degradation\_P}$ (representing the kinetic law for the protein degradation), and some parameter values in the set $\mathcal{K}$.

$$P \stackrel{def}{=} (translation, 1)\uparrow P + (dimerization, 2)\downarrow P + (monomerization, 2)\uparrow P;$$

$$Res \stackrel{def}{=} (degradation\_M, 1) \odot Res;$$

$$(((D(1) \underset{\{transcription\}}{\bowtie} M(0)) \underset{\{translation\}}{\bowtie} P(0)) \underset{\{dimerization,monomerization\}}{\bowtie} P2(0)) \underset{\{degradation\_M\}}{\bowtie} Res(1)$$

The new set of parameters $\mathcal{K}'$ is:

$$K_{\mathrm{M}} = 356 \text{ molecules}; \quad v = 2.19 \text{ s}^{-1}; \quad k_2 = 0.03 \text{ s}^{-1};$$
$$k_3 = 0.039 \text{ s}^{-1}; \quad k_5 = 0.06 \text{ s}^{-1}; \quad k_{-5} = 0.5 \text{ s}^{-1} \ .$$

With respect to $\mathcal{M}_1$, the rate of dimerization (k_5) is increased, the rate of translation (k_2) is decreased, and k_4 is removed as we assume that there is no protein degradation.

5.2 Simulation and Model-Checking

Here we apply our approach to both $\mathcal{M}_1$ and $\mathcal{M}_2$. Notice that since both describe an unbounded network, it is not possible to calculate even loose theoretical bounds from the initial conditions.

5.2.1 Network $\mathcal{M}_1$

We focus first on $\mathcal{M}_1$. We perform 1000 independent stochastic simulation runs using Gillespie's Direct Method [8]. The chosen number of runs is large enough to take into account the variability of the system, but still makes the total simulation time reasonable (in the order of minutes). We used $T = 20000$ *s* as a simulation stop-time: by that time the system has reached a stable state.

The simulation results are reported in Fig. 4, which shows the average values obtained over all the runs. As the figure shows, both the monomeric protein P and the dimeric protein $P2$ rapidly increase until they reach levels which remain stable within the considered time

bound.

This simple network is an interesting example because, despite being structurally unbounded, a stable behaviour is observed by looking at the average values of multiple runs: the amount of all the species is not unbounded.

We can estimate the upper bounds for the amounts of each species as the maximum values obtained in any run at any time instant,

$$Max_M = 5; \quad Max_P = 33; \quad Max_{P2} = 18$$

and we can use these values in the PRISM model. The amounts of the other species, D and Res, are not affected by the reactions in which they are involved and, therefore, they are constant at value 1.

In this example, the lower bounds for all the interesting species are 0, since they are not present at system initialisation.

In Fig. 4 we also report the expected values for the amounts of the interesting species (M, P and $P2$) obtained by PRISM using the derived bounds. The results are in agreement with the average values calculated from the simulation runs. These values have been obtained by checking the instantaneous reward properties

$$\mathcal{R}^{M}_{=?}[\mathrm{I} = t], \quad \mathcal{R}^{P}_{=?}[\mathrm{I} = t], \quad \mathcal{R}^{P2}_{=?}[\mathrm{I} = t]$$

varying the time $t \in [0, T]$, where M, P and $P2$ are reward structures associating with each state the current number of molecules of each species. In the same Fig. 4 we consider the standard deviation of the number of molecules for the PRISM model. Specifically, we define reward structures associating with each state the square of the number of molecules of each species and the standard deviation is then calculated as the square root of the variance $E(Y)^2 - E(Y^2)$, where Y is the random variable representing a species in the network, whereas $E(Y)$ and $E(Y^2)$ indicate the expected values for the amount of the species Y and for its square value.

In order to confirm the belief that the choice of the bounds is crucial for having the right compromise between correctness and efficiency, we performed a few more experiments, both with smaller and with bigger upper bounds. The result is that, if the selected bounds are too low, the obtained behaviour is not in agreement with the average simulation result. On the other hand, using bounds which are too high has the effect of dramatically increasing the state space and, thus, it makes the model-checking much slower. For instance, the verification of the next set of properties in a model where the bounds for the three species are doubled is over 20 times slower, while verifying the previously-used CSL formulae (with no increase in the bound to be reached to satisfy the formula) produces the very same values reported in Fig. 5. All values are equal up to the fifth decimal digit at least. This confirms that the bounds we have imposed do not alter the behaviour of the system.

The agreement of the stochastic simulation results and the expected values computed by model-checking is a form of validation of our approach per se: it shows that we have not introduced significant errors. As another form of validation of the derived bounds, we have calculated the probabilities of reaching them at different time instants ($\mathcal{P}_{=?}[true\ \mathrm{U}^{\leq t}\ M = 5]$, $\mathcal{P}_{=?}[true\ \mathrm{U}^{\leq t}\ P = 33]$, and $\mathcal{P}_{=?}[true\ \mathrm{U}^{\leq t}\ P2 = 18]$). The results, reported in Fig. 5, provide a means for estimating the error which might have been introduced by bounding the system.

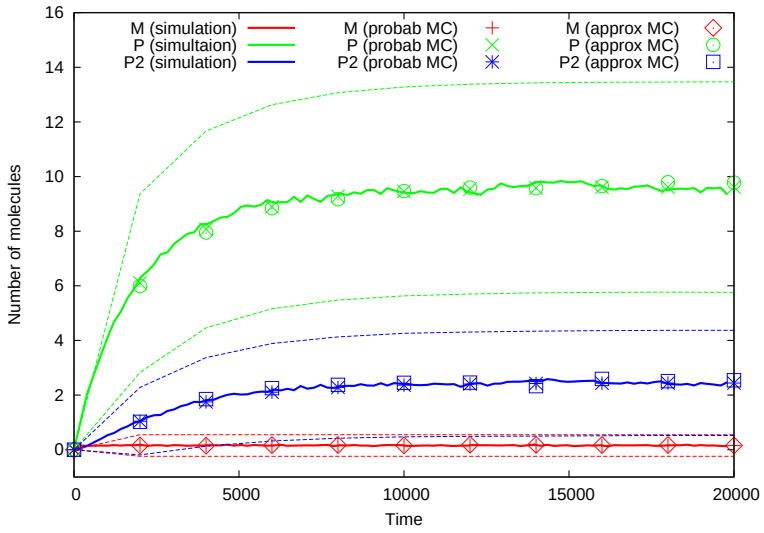


Fig. 4. Simulation, exact and approximate probabilistic model-checking results for $\mathcal{M}_1$: average number of molecules over 1000 simulation runs (thick lines), expected number of molecules at time t (points) and its standard deviation (thin dashed lines), obtained by using reward structures in PRISM. We observe that the agreement between the three sets of results is good.

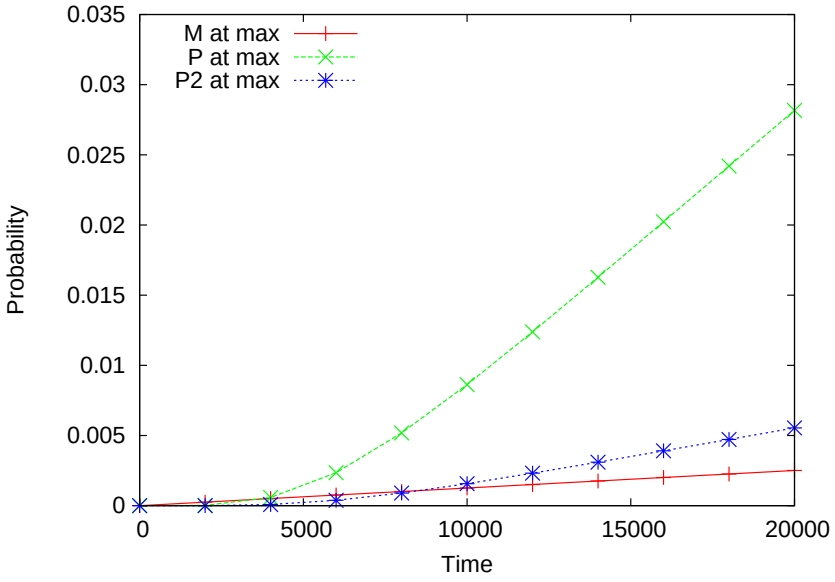


Fig. 5. Model-checking results for $\mathcal{M}_1$: probability of each species reaching its upper bound.

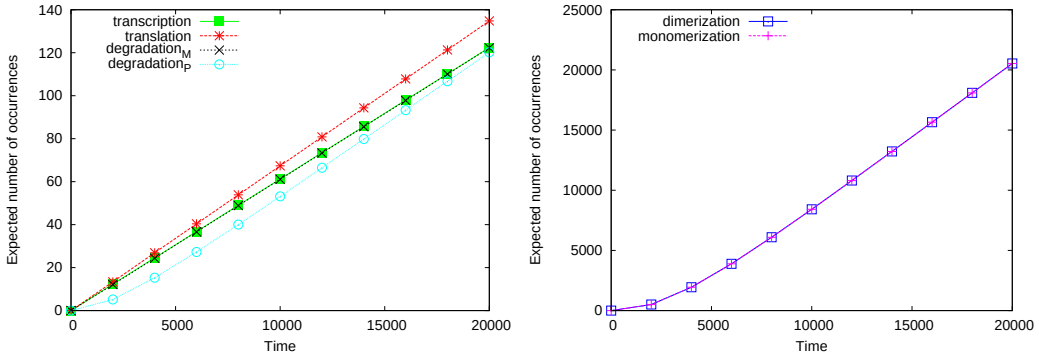
In this case the state space truncation has almost no impact on the behaviour of the system. However, it is worth mentioning that, in general, the effect of such truncation is dependent on the nature of the specific system under study. Notably, the results obtained by checking the last property shown can be used to refine the model, by increasing the bounds in case the probability of them being reached is considered to be too high. When one is satisfied with the resulting probabilities, PRISM can be used to evaluate other CSL formulae on the defined model, in order to provide additional insight into the system behaviour.

We consider in the following a number of properties which can be automatically checked, though for such a small example the behaviour of the system is simple and the model-checking approach does not happen to be particularly meaningful. In the case of real life systems with complex behaviour further specific properties could be verified.

As a first example we consider the cumulative reward property

$$\mathcal{R}^{\langle react \rangle}_{=?}[C \leq t]$$

where $\langle react \rangle$ is a reaction name (e.g. "transcription", "translation", "degradation_M", etc.), with which a transition reward is associated. These properties, analysed at different time instants $t \in [0, T]$ return the expected number of occurrences of a reaction by that time (see Fig. 6, where we have separated slow and fast reactions in two different graphs for the sake of readability because of their very different scales).



(a) slow reactions (b) fast reactions

Fig. 6. Model-checking results for $\mathcal{M}_1$: expected number of occurrences of reactions by time t.

Fig. 7 shows the expected amounts of monomers (P) compared to the total amount of proteins ($P + P2$) present at time t. Specifically, it refers to the formula $\mathcal{R}^{ratio}_{=?}[\mathrm{I} = t]$ where "ratio" is a reward structure which associates the reward $\frac{P}{P+P2}$ with each state.

We point out that properties such as the ones just mentioned and many others (e.g. the expected time at which a protein dimer is first produced, the probability of having more dimers than monomers at some time t, etc.) can be evaluated in a probabilistically precise manner by model-checking: all the user needs to do is to provide the model-checker with the CSL specification of the desired property and wait for the definitive answer. On the other hand, the analysis of these properties by simulation, if possible, requires the use of reward-based simulation tools in which the user often needs to implement the reward functions themselves; in addition, the answers regarding the satisfaction of properties obtained by simulation could only be up to some confidence interval and, often, an extremely high number of simulation runs would be required to achieve a satisfactory confidence.

In this example, we have checked all properties at time instants in the interval $[0, T]$. However, as one can easily see from Fig. 4, the system reaches a stable behaviour much before time T. In cases like this, and if the time required for model-checking experiments is long, the number of experiments to run could be reduced by considering this further information from the simulation results.

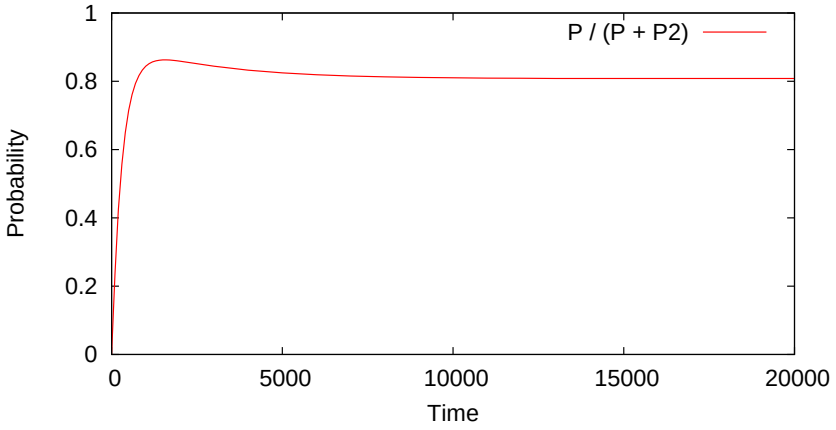


Fig. 7. Model-checking results for $\mathcal{M}_1$: expected ratio between protein monomers and total proteins at time t.

5.2.2 Network $\mathcal{M}_2$

We consider here $\mathcal{M}_2$. The simulation results reported in Fig. 8 show that the system does not reach a stable (average) state within the simulation time in which we are interested ($T = 20000s$): since in the model there is no protein degradation, both the monomeric protein P and the dimeric protein $P2$ increase without limit.

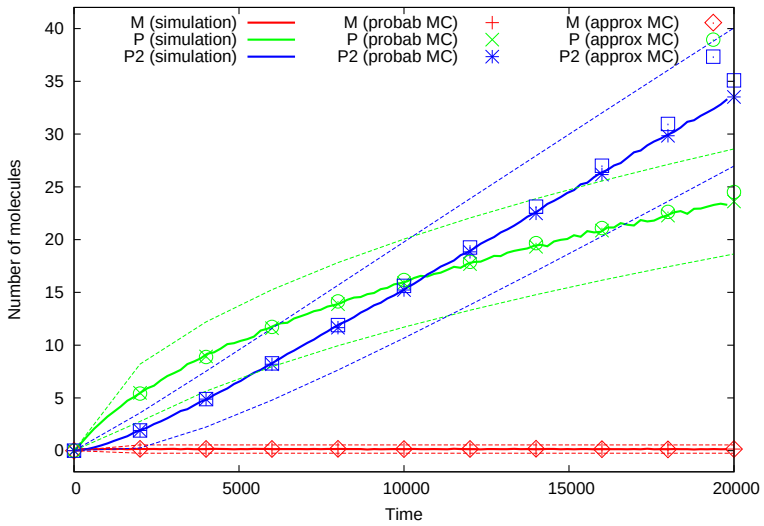


Fig. 8. Simulation, exact and approximate probabilistic model-checking results for $\mathcal{M}_2$: average number of molecules over 1000 simulation runs (thick lines), expected number of molecules at time t (points) and its standard deviation (thin dashed lines) obtained by using reward structures in PRISM. We observe that the agreement in all three cases is good but that the strongest agreement is between the average simulation results and the probabilistic model-checking results. The results obtained by approximate model-checking appear to indicate higher numbers of both monomers and dimers than the other two methods.

Again, we estimate the upper bounds for the amounts of each species as the maximum values obtained in any run at any time instant up to the simulation stop-time,

$$Max_M = 5; \quad Max_P = 51; \quad Max_{P2} = 64$$

and we use these values in the PRISM model. The amounts of D and Res are constant in this case, too.

In Fig. 8 we report the expected values (and their standard deviation) of the amounts of the interesting species obtained by PRISM, which are in agreement with the average values calculated from the simulation runs. These values have been obtained by checking the instantaneous reward properties

$$\mathcal{R}^{M}_{=?}[\mathrm{I} = t], \quad \mathcal{R}^{P}_{=?}[\mathrm{I} = t], \quad \mathcal{R}^{P2}_{=?}[\mathrm{I} = t] \ .$$

Fig. 9 reports on the probabilities of reaching the upper bounds at different time instants ($\mathcal{P}_{=?}[true\ \mathrm{U}^{\leq t}\ M = 5]$, $\mathcal{P}_{=?}[true\ \mathrm{U}^{\leq t}\ P = 51]$, and $\mathcal{P}_{=?}[true\ \mathrm{U}^{\leq t}\ P2 = 64]$).

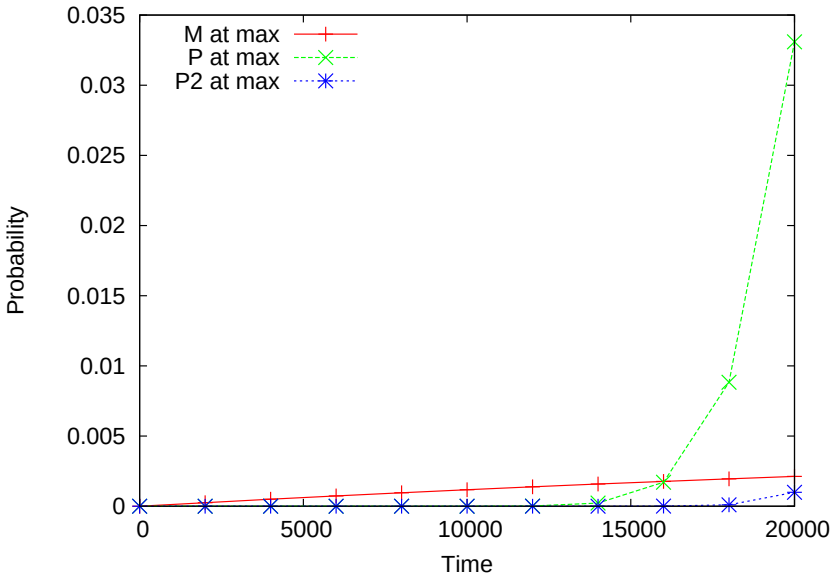
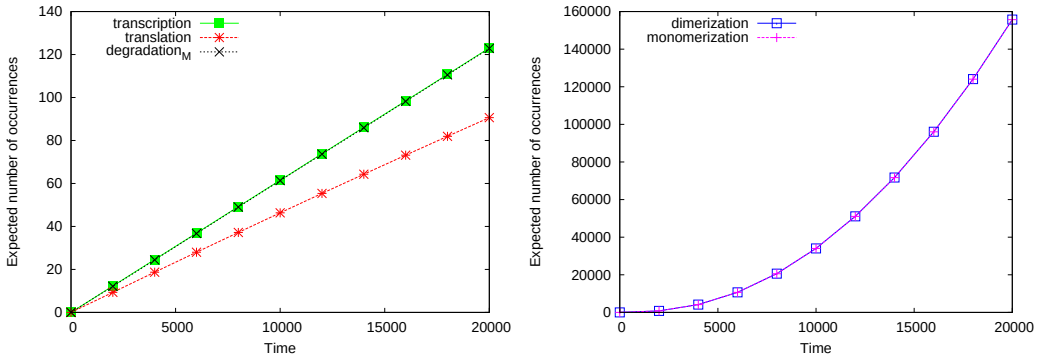


Fig. 9. Model-checking results for $\mathcal{M}_2$: probability of each species reaching its upper bound.

The occurrences of each reaction by time t are reported in Fig. 10, while Fig. 11 shows the expected amounts of monomers (P) compared to the total amount of proteins ($P + P2$) present at time t (they refer to the same properties described for $\mathcal{M}_1$).

Finally, Fig. 12 shows the probability of the amount of $P2$ being greater than the amount of P at different time instants ($\mathcal{P}_{=?}[true\ \mathrm{U}^{[t,t]}\ P2 > P]$). As expected, this probability increases as time increases. However, it does not reach 1. Indeed, even if the average behaviour shows that in the long run the amount of $P2$ is greater than the amount of P (see Fig. 8), this is not generally true for all possible behaviours: the system is highly variable, and there can be some paths in which $P2 < P$ even when approaching the simulation stop-time. This is not very surprising because the confidence intervals around the times series for P and $P2$ are still overlapping up to the stop time of 20000 seconds (see Fig. 8). The formal confirmation of this remark is given by the fact that the probability of $P2$ being



(a) slow reactions (b) fast reactions

Fig. 10. Model-checking results for $\mathcal{M}_2$: expected number of occurrences of reactions by time t.

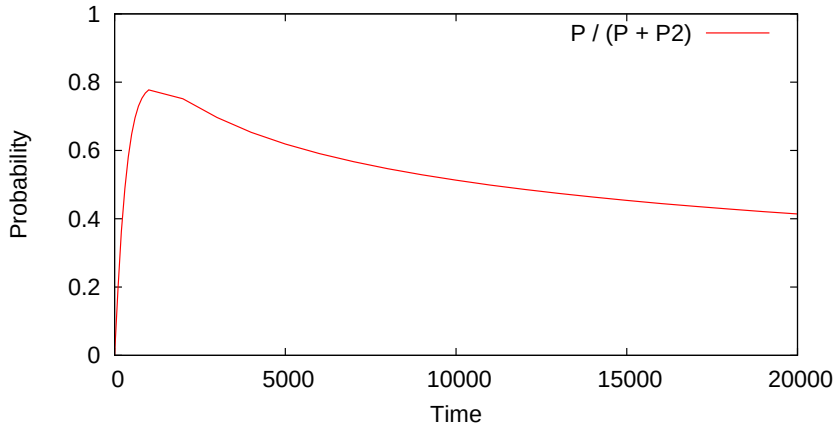


Fig. 11. Model-checking results for $\mathcal{M}_2$: expected ratio between protein monomers and total proteins at time t.

greater than P in every state after time t ($\mathcal{P}_{=?}[\mathrm{G}^{[t,T]}\ P2 > P]$) is 0 for any $t \in [0,T)$.

6 Related Work

In this work, as in our earlier work [4], we are concerned with obtaining the exact probability distribution across all of the states of the reachable state-space of a network of chemical reactions. In our earlier work we used the stochastic process algebra PEPA [15,14] to express the model and applied numerical linear algebra to solve the underlying Markov chain. In this paper we are using the Bio-PEPA language and applying model-checking to obtain our results.

The use of probabilistic model-checking for the analysis of models of biological phenomena is already well established. In [5] the authors consider signal transduction in the RKIP-inhibited ERK pathway and manage the state-space explosion problem by using approximate techniques where concentrations are modelled by discrete abstract quantities.

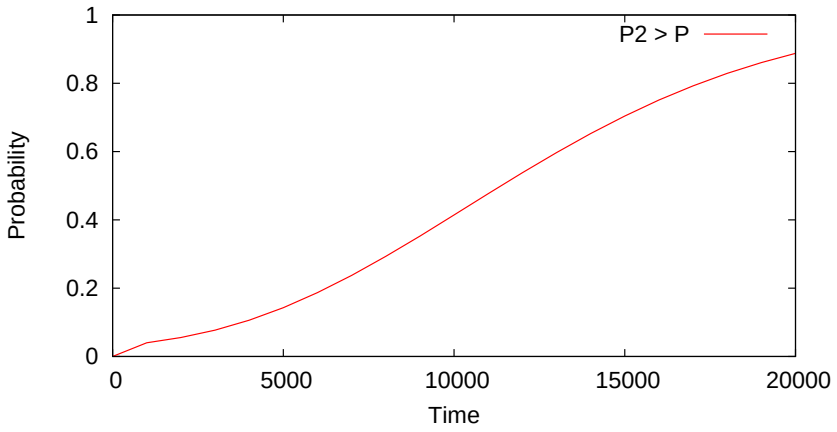


Fig. 12. Model-checking results for $\mathcal{M}_2$: expected probability of $P2 > P$ at time t.

In [9] the authors apply model-checking to the complex FGF (Fibroblast Growth Factor) signalling pathway.

Another programme of work bringing together the analysis methods of stochastic simulation and the numerical solution of the Markov chain is *perfect sampling* [10]. This work joins the "Dominated Coupling From The Past" Algorithm from Monte-Carlo Markov Chain theory with Gillespie's Stochastic Simulation Algorithm in the DCFTP-SSA [11]. The DCFTP-SSA guarantees sampling from the stationary probability distribution of the Chemical Master Equation and can be used to study steady-state properties of a broad class of stochastic biochemical networks. Our work guarantees that we are using the transient probability distribution of the CME and can be used to study time-dependent properties of a broad class of stochastic biochemical networks up to the stop-time of the simulation pre-run.

7 Conclusions

Summing up, our approach is the following:

- We consider a Bio-PEPA system $\mathcal{M}$ representing a biochemical network, and we automatically derive from it a model specification to be used for stochastic simulation (by Dizzy) and one to be used for model-checking (by PRISM).
- We set the simulation time T and the number of simulation runs.
- We pick as bound for a species the largest number of molecules which that species has obtained in any simulation run within time T.
- We update the PRISM model derived from the Bio-PEPA model with the estimated bounds, and we validate this model by comparing the expected values calculated by PRISM with the average values obtained by simulation.
- We use PRISM to analyse the model by verifying specific CSL properties.

In addition to the fact that simulation allows us to set some bounds which make model-checking feasible, the combination of those two analysis techniques is itself advantageous. Simulation and model-checking are complementary techniques and they can be used to investigate different properties of the same system in order to give a more complete understanding. Moreover, since the Dizzy model and the PRISM model are equivalent, we expect the results obtained with the two approaches to be in agreement. If this is the case, that can give us staunch confidence about the correctness of the results. If, instead, we get different results, that means there is some mistake in either approach (e.g. more simulation runs need to be performed, the chosen bounds are too low, or the simulation stop-time is too small), and this information could be used to refine the model or the simulation settings.

We have automated the generation of the simulation model and the model-checker input from a model expressed in the Bio-PEPA process algebra. We have automated the repeated execution of a set of independent simulation runs and the identification of the maximum and minimum values of each species from the simulation results. The user then only needs to load the PRISM model, supply the identified parameters and execute it. However, the choice of the number of simulation runs to be performed remains in the hands of the modeller.

We point out that the upper bounds estimated by using this approach are approximate because, given the stochastic nature of the simulation, we can have genuine confidence in the chosen bounds only if we run a suitably large number of simulations. However, this issue is not specific to this approach. The choice of the number of simulation runs is needed for any stochastic simulation experiment; on top of that, we use the simulation just as a supporting technique to the model-checking, and we believe that the combined use of the two approaches helps to minimize the uncertainty due to the stochastic simulation.

The good agreement between the results obtained by simulation and by model-checking on the presented example makes us confident that, provided an adequate number of simulation runs is performed, then our approach does not introduce significant errors. However, we stress again the fact that the sensitivity to the truncation of the state space is strongly dependent on the system itself; therefore, in order to assess the correctness of the estimated bounds for a specific system, the results obtained by model-checking should be validated against the behaviour obtained by simulation and against previous experimental and computational data, if there is any available.

Acknowledgement

The authors thank the anonymous reviewers of this paper for their insightful comments on the work. Federica Ciocchetta and Jane Hillston are supported by the U.K. Engineering and Physical Sciences Research Council (EPSRC) Advanced Research Fellowship and research grant EP/C543696/1 “Process Algebra Approaches to Collective Dynamics”. Stephen Gilmore and Maria Luisa Guerriero are supported by the EPSRC grant EP/E031439/1 “Stochastic Process Algebra for Biochemical Signalling Pathway Analysis”.

References

[1] Aziz, A., K. Kanwal, V. Singhal and V. Brayton, *Verifying continuous time Markov chains*, in: *Proc. 8th International Conference on Computer Aided Verification (CAV'96)*, LNCS **1102** (1996), pp. 269–276.

[2] Baier, C., J.-P. Katoen and H. Hermanns, *Approximate Symbolic Model Checking of Continuous-Time Markov Chains*, in: *Proceedings of CONCUR'99*, LNCS **1664**, 1999, pp. 146–161.

[3] Bundschuh, R., F. Hayot and C. Jayaprakash, *Fluctuations and Slow Variables in Genetic Networks*, Biophys. J. **84** (2003), pp. 1606–1615.

[4] Calder, M., S. Gilmore and J. Hillston, *Modelling the influence of RKIP on the ERK signalling pathway using the stochastic process algebra PEPA*, Transactions on Computational Systems Biology VII **4230** (2006), pp. 1–23.

[5] Calder, M., V. Vyshemirsky, D. Gilbert and R. Orton, *Analysis of signalling pathways using continuous time Markov chains*, Transactions on Computational Systems Biology VI **4220** (2006), pp. 44–67, springer.

[6] Ciocchetta, F. and J. Hillston, *Bio-PEPA: a framework for the modelling and analysis of biological systems* (2008), School of Informatics University of Edinburgh Technical Report EDI-INF-RR-1231.

[7] Ciocchetta, F. and J. Hillston, *Bio-PEPA: an extension of the process algebra PEPA for biochemical networks*, in: *Proc. of FBTC 2007*, Electronic Notes in Theoretical Computer Science **194**, 2008, pp. 103–117.

[8] Gillespie, D., *Exact stochastic simulation of coupled chemical reactions*, Journal of Physical Chemistry **81** (1977), pp. 2340–2361.

[9] Heath, J., M. Kwiatkowska, G. Norman, D. Parker and O. Tymchyshyn, *Probabilistic Model Checking of Complex Biological Pathways*, Theoretical Computer Science **319** (2008), pp. 239–257, special Issue on Converging Sciences: Informatics and Biology.

[10] Hemberg, M. and M. Barahona, *Perfect sampling of the master equation for gene regulatory networks*, Biophys J. **93** (2007), pp. 401–10.

[11] Hemberg, M. and M. Barahona, *A Dominated Coupling From The Past algorithm for the stochastic simulation of networks of biochemical reactions*, BMC Syst Biol. **2** (2008), pp. 401–10.

[12] Hérault, T., R. Lassaigne, F. Magniette and S. Peyronnet, *Approximate probabilistic model checking*, in: *Proceedings of the 5th Verification, Model Checking and Abstract Interpretation (VMCAI 2004)*, LNCS **2937**, Venice, Italy, 2004, pp. 73–84.

[13] Hérault, T., R. Lassaigne and S. Peyronnet, *APMC 3.0: Approximate Verification of Discrete and Continuous Time Markov Chains*, in: *Proceedings of the 3rd International Conference on the Quantitative Evaluation of SysTems (QEST'06)*, California, USA, 2006.

[14] Hillston, J., *Process algebras for quantitative analysis*, in: *Proceedings of the 20th Annual IEEE Symposium on Logic in Computer Science (LICS' 05)* (2005), pp. 239–248.

[15] Hillston, J., *Tuning systems: From composition to performance*, The Computer Journal **48** (2005), pp. 385–400, the Needham Lecture paper.

[16] Hinton, A., M. Kwiatkowska, G. Norman and D. Parker, *PRISM: A tool for automatic verification of probabilistic systems*, in: H. Hermanns and J. Palsberg, editors, *Proc. 12th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'06)*, LNCS **3920** (2006), pp. 441–444.

[17] Ramsey, S., D. Orrell and H. Bolouri, *Dizzy: stochastic simulation of large-scale genetic regulatory networks*, J. Bioinf. Comp. Biol. **3** (2005), pp. 415–436.

[18] Rathinam, M., L. Petzold, Y. Cao and D. Gillespie, *Stiffness in stochastic chemically reacting systems: The implicit tau-leaping method*, Journal of Chemical Physics **119** (2003), pp. 12784–12794.